Remarks

Entry of the amendments, reconsideration of the application, as amended, and allowance of all pending claims are respectfully requested. Claims 1-9, 11-19 and 23-31 remain pending.

Applicants gratefully acknowledge the time afforded applicants' representative, Blanche E. Schiller, during a telephonic interview on April 25, 2005, in which claim 1 and the cited art, Deitel and Burk, were discussed. During the interview, applicants explained that the search tool of one aspect of the present invention is for use outside of the compile phase of the program, such as when debugging the program after the compiler detects an error. Applicants agreed to amend the claims accordingly. The Examiner indicated that a further search would need to be performed and no agreement with respect to the claims was reached.

In furtherance of the Examiner's interview and of this application, applicants have amended the claims to specifically indicate that the search tool of an aspect of the present invention is used in debugging computer programs outside of a compile phase of the computer program. That is, applicants' invention is directed to search tools that are used outside of compiling. Support for the amendments can be found in applicants' specification. For example, search tools that have deficiencies addressed by one or more aspects of applicants' invention are specifically mentioned in paragraph 2 of the Background (page 1). Grep and Find are search tools known in the art to perform searching outside of the compiler or compile phase. In publications describing Grep, for instance, there is no mention of using Grep during the compile process (see, e.g., http://www.linuxjournal.com/article/1149; http://www.opengroup.org/onlinepubs/009695399/utilities/grep.html; and http://www.regular-expressions.info/grep.html). Conversely, C programming books fail to mention Grep (see, e.g., IBM C/C++ for MVS/ESA C/MCS Language Reference, SC09-2063-00; IBM C/C++ for MVS/ESA: C/MVS Programming Guide, SC09-2062-00; z/OS V1R 1.0 and OS/390 V2R6.0-V2R10.0 C/C++ IBM Open Glass Library Reference, SC09-2364-04; IBM C/C++ for MVS/ESA: C/MVS Reference Summary, SX09-1303-00; IBM C/C++ for MVS/ESA: C/MVS User's Guide, SC09-2061-00; z/OS V1R2.0-V1R4.0 C/C++ Messages, GC09-4819-00; and OS/390 V2R9.0 C/C++ Language Reference, SC09-2360-04). Thus, applicants respectfully

submit that one skilled in the art would understand that search tools, like Grep and Find, mentioned in applicants' invention perform searching outside of the compile phase of a program.

Applicants' Background describes the shortcomings of Grep and Find and how enhancements are needed for such search tools. Additionally, paragraph 36 indicates that the logic of the present invention can be used in a search tool. Further, search logic or tool is described in paragraph 18 of the specification. Paragraph 22 specifically cites instructions that are typically used during a compiler and are now used with aspects of the present invention to target or limit a search. Yet, further, the debugging of a program is described in paragraph 16 on page 5 of the specification. Based on the foregoing, applicants respectfully submit that no new matter is being added by these amendments.

Applicants' invention is directed, in one aspect, to improving search tools, such as Grep and/or Find, by providing targeted searching in order to improve the accuracy and speed of the search. As is known, search tools are used, for example, to facilitate debugging programs, subsequent to, for instance, a compiler indicating an error. To enhance the search tools, the search is limited, in accordance with an aspect of the present invention, to specific items (e.g., files) associated with the program of the search. That is, other files in, for instance, the same directory being searched as the associated files are not included in the search. In a further enhancement, the search is limited to specific items in specific locations (e.g., directories associated with the program). Again, other directories are not searched.

This targeted search evolves, however, based on language specific rules of the program. The language specific rules are used to determine if additional items should be searched. For example, if an #include statement is found while performing the search, then the search continues in that included item.

In one particular aspect, applicants claim a method of facilitating searching for text of computer programs (e.g., independent claim 1). The method includes, for instance, identifying text of a computer program to be searched by a search tool used in debugging the computer program outside of a compile phase of the computer program; searching for the text by the search tool in a targeted item at a targeted location associated with the computer program; determining whether one or more additional items are to be searched for the text, the determining

using one or more language specific rules of the computer program to determine whether one or more additional items are to be searched, wherein the determining is defined by the computer program; and searching for the text by the search tool in at least one additional item of the one or more additional items, in response to the determining indicating that one or more additional items are to be searched. Thus, in this aspect of applicants' claimed invention, a search tool is used to search text of a computer program. That search tool is being used in debugging the computer program outside of a compile phase of the computer program. The search tool searches for the text in items that are determined using one or more language specific rules of the computer program. Such a search tool is not described, taught or suggested in any of the cited references, either alone or in combination.

For instance, while Burk describes Find and Grep commands, which are used to search files for selected text, the Find and Grep commands of Burk are very different from the search tool of applicants' claimed invention. Applicants' claimed invention provides an enhanced search tool that overcomes the deficiencies of Find and Grep. For instance, Find and Grep, as described in Burk, require a user to specify the search criteria, which is quite different from applicants' claimed invention, in which it is the computer program itself (e.g., the language specific rules of the computer program) that defines where the search is to be performed. Take the following C or C++ program:

SAMPLE PROGRAM

```
#INCLUDE<myclass.h>
#INCLUDE<stdio.h>

INT MAIN () {
PRINT ("hello world\n"):
VARIABLE X = 17;
FUNCTIONCALLY=();
}
```

As one example, assume the sample program is executed on a machine having the following environment variable:

INCLUDE=C:\IBMCPP\INCLUDE;C:\CLASSLIB\INCLUDE;
E:\RICK\INCLUDE;

In this example, the identified text (e.g., FunctionCallY) is only searched, in accordance with an aspect of the present invention, in the selected files (e.g., source.c, myclass.h and stdio.h) in the following chosen directories: C:\IBMCPP\INCLUDE; C:\CLASSLIB\INCLUDE; and E:\RICK\INCLUDE. Other files and/or directories are not searched.

The search is defined by the #include statements of the computer program, rather than by a user specifying search criteria. By having the computer program define the search, the search may be revised (e.g., expanded) to include other targeted items or locations to be searched, as the search progresses. This is very different from Burk, as well as the Grep and Find commands.

Grep and Find allow user defined arguments to be specified to help limit the scope of what is searched, such that the results are narrowed. However, Grep and Find have no way of enlarging the scope of the search by the computer program itself, as in applicants' claimed invention. Because the Grep and Find tools have no real concept of what they are searching, other than it being raw data, Grep and Find start with a large amount of objects to search and use arguments to limit which objects are searched. In contrast, applicants' claimed invention starts with a small set of objects and expands it to encompass other small sets through understanding of where to look based on knowledge of the object it is searching. This defining is by the computer program itself.

If Find or Grep was used to search on a method or function, the search would result in many incorrect matches, because neither Find nor Grep understands the difference between a function/method call and a function/method declaration or even a function/method prototype. Furthermore, if Find or Grep was used to find documentation on the same function/method, it would not understand the man pages or help files or pdfs or html to know how to find just the place that it is documented (no concept of indexes). It would show all of the references to function/method, as well as any supporting documentation. This is very different from applicants' claimed invention, in which the search is targeted and defined by the program to reduce false positives.

Based on the foregoing, applicants respectfully submit that Burk does not describe, teach or suggest applicants' claimed invention.

Further, Deitel does not overcome the deficiencies of Burk. Deitel is directed to preprocessing and compiling, and not to search tools and techniques, like Grep, Find and applicants' claimed invention, which are used outside of the compile phase of a computer program, such as in debugging the program.

Deitel describes searching during compile time. Deitel does not extend to environments outside of the compile phase, such as to search tools, like Grep and Find, used, for example, during debugging subsequent to the compiler finding an error. It is with these search tools that applicants have identified a need. It is with the non-compiler search tools that applicants have found the searching to be inadequate. Applicants have improved these types of search tools to enhance the searching thereof. Although compilers have been around for many years, it has never been thought to combine the teachings of a compiler with the techniques of search tools that are used outside of the compile phase. There is no teaching or suggestion in the references or elsewhere to combine compiler search techniques with the techniques of search tools used outside of the compile phase. Such a combination is hindsight reconstruction of applicants' invention.

Since Burk is directed to non-compiler phase search tools, but Deitel is directed to the compiler, and there is no teaching or suggestion in the references to combine the techniques used in different processing phases, applicants respectfully submit that their invention is patentable over the combination of Burk and Deitel.

For all of the above reasons, applicants respectfully request an indication of allowability for independent claim 1, as well as the other independent claims. Further, the dependent claims are patentable for the same reasons as the independent claims, as well as for their own additional features.

If the Examiner believes any issue could be expeditiously resolved through a telephonic interview, the Examiner is invited to call applicants' undersigned representative.

Respectfully submitted,

Blanche E. Schiller

Blanche E. Schiller
Attorney for Applicants
Registration No.: 35,670

Dated: May 23, 2005.

HESLIN ROTHENBERG FARLEY & MESITI P.C.
5 Columbia Circle
Albany, New York 12203-5160
Telephone: (518) 452-5600
Facsimile: (518) 452-5579